Ad Hoc Networks 88 (2019) 5–17

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Efficient broadcast in opportunistic networks using optimal stopping theory

Carlos Borrego*, Joan Borrell, Sergi Robles

Department of Information and Communications Engineering, Universitat Autònoma de Barcelona, Bellaterra, Spain

ARTICLE INFO

Article history:
Received 2 May 2018
Revised 15 November 2018
Accepted 2 January 2019
Available online 5 January 2019

Keywords:
Opportunistic Networks
Broadcasting
Message dissemination
Optimal stopping theory

ABSTRACT

In this paper, we present a broadcast dissemination protocol for messages in opportunistic networks (OppNet) that is efficient in terms of energy consumption and network capacity usage, while not increasing the number of excluded nodes (nodes not receiving messages). The majority of the OppNet broadcast delivery schemes proposed in the literature, do not take into consideration that reducing energy and buffer usage is of paramount importance in these wireless networks normally consisting of small devices. In our protocol, broadcast messages are limited by carefully selecting their prospective forwarders (storers). The keystone of our protocol is the use of Optimal Stopping Theory, which selects the best message storers at every stage of the algorithm, while holding back broad message dissemination until convenient conditions are met. The broadcast efficiency of the proposed protocol out competes other OppNet broadcast proposals in four well-known scenarios. Furthermore, the protocol reduces the number of both dropped messages and nodes not receiving messages, thus maximising network capacity usage, and the span of the message delivery.

1. Introduction

When nodes need to communicate wirelessly, they can form an ad-hoc network using the links of direct neighbours to reach further nodes or can use a managed wireless network (assisted by some infrastructural elements such as access points). However, sometimes nodes cannot be reached using these networks, due in part to node mobility or availability reasons, and an alternative means of communication, such as opportunistic networking, is needed. In opportunistic networks (*OppNet*) [14], messages are sent to nodes in direct communication range, who store and carry these messages until there is a new opportunity to forward the messages to other nodes. This provides a global communication service even when message storers only connect to other nodes temporarily. OppNets are not contenders for the fastest or most reliable network, but are second to none regarding delay and disruption tolerance. This makes them a good candidate in a number of scenarios, including low cost —energy, resources, money— networks such as sensor networks, or the Internet of Things; interplanetary or submarine networks; and networks where nodes need to maintain the privacy of their whereabouts or the anonymity of their users. Although OppNet represent a communication solution where other networks can be hardly used, there are still issues to be addressed before they can be widely deployed, such as efficient routing and delivery.

Routing and delivering of messages have been traditionally considered two sides of the same coin. This is, in part, because in connected networks, the recipients of the messages and the way they have to be forwarded are closely coupled. Examples of this are IP Multicast and IP Anycast. When it comes to opportunistic networks, the bond between routing and delivery fades away. Which nodes are intended to receive messages could be totally unrelated to the way these messages are routed through the network. An example where this can be easily seen is broadcast: in OppNet, although a message has to arrive at every single node in the network (broadcast delivery), there are many ways of routing it (broadcast routing), such as flooding, sending a limited number of copies, or carefully selecting the forwarding nodes –also called storers.

The use of unicast delivery (messages intended to a particular node) in OppNet is generally set aside in favour of a broadcast-like delivery. This is mainly due to the very nature of the network, in which opportunistic node gatherings make it difficult to have an efficient routing for these messages in arbitrarily large networks with discretionary node addresses. On the other hand, broadcast delivery is widely used in OppNets, and having an efficient routing strategy for these broadcast messages is essential for the imple-

* Corresponding author.
E-mail addresses: carlos.borrego@uab.cat (C. Borrego), joan.borrell@uab.cat (J. Borrell), sergi.robles@uab.cat (S. Robles).

https://doi.org/10.1016/j.adhoc.2019.01.001
1570-8705/© 2019 Elsevier B.V. All rights reserved.

mentation of useful real-world applications. One might think that this is a problem easy to solve. A simplistic approach to broadcast in opportunistic networking would be to have all nodes sending the broadcast message to all nodes at sight, and perhaps keep transmitting the same message to other nodes in the future as well (so becoming a storer). The problem would be a general flooding of the network, with not even the guarantee that all nodes would receive a copy. In terms of efficiency, this strategy is therefore definitely not one of the best. There are two important aspects that have to be considered here: the carry involved in OppNet emphasises the importance of the limitation posed by the network "capacity", and hence a constraint to be considered; and as these devices tend to have limited battery capacity, such as IoT sensors, energy consumption has to be restricted, which involves limiting the number of transmissions as much as possible. Hence, designing a coherent strategy to broadcast messages in OppNet is a complex task that has to take into account these constraints and ensure, at the same time, delivery efficiency in terms of dissemination, delivery time, and energy cost.

In the literature, some OppNet broadcast approaches have been proposed [19,40] but typically these are based on limiting the number of message copies or their lifespan, and are oblivious to the impact on network capacity in the overall performance, which leads to unwanted consequences such as uneven message dissemination or having unreachable areas and excluded nodes.

In this paper, we offer a different perspective to the problem of message broadcasting in OppNet. We propose that broadcast messages should be limited but not just regarding the number of copies or their lifespan, but also by carefully select their prospective forwarders (storers). Nodes that receive a message can accept it and delete it; store and carry the message, delivering it to other nodes in future encounters –which will delete the message after acceptance;– or pass it to nodes that will be selected as new forwarders. These forwarding nodes, named storers, are selected in terms of three different criteria: centrality –how well connected a node is in the network–, reliability –the likeliness a node does not drop messages–, and similarity –how alike a node is in terms of shared acquaintances–. The goal is to make the broadcast effective, even, and pervasive all over the network using the minimum energy possible, while reducing the number of excluded nodes.

The main contribution of this paper is a broadcast dissemination protocol for messages in an OppNet, that is efficient with respect to message latency and message dissemination per unit of energy. The protocol, presented in Section 3, keeps low the percentage of excluded nodes (nodes not receiving messages), while trying to maximize the range of message delivery. The keystone of the protocol is the use of Optimal Stopping Theory (OST), introduced in Section 2, to select the best message storers in each stage of the algorithm, holding back broad message dispersal until convenient conditions are met. Even though the solution of finding the storers in each stage of our algorithm results in optimality, global optimisation (i.e. choosing the best storers possible for a message) is not pursued in this proposal. The novelty of our proposal is the idea that these message storers are the only nodes allowed to perform the whole store-carry-and-forward process. This strategy reduces the number of dropped messages and thus makes the most of the network storage capacity, facilitating the spreading of messages throughout the network. The metrics used to assess favourable forwarding conditions include betweenness egocentric centrality; least dropping probability and most node capacity to store messages; and the least common contacted nodes. The broadcast efficiency of the protocol out performs other OppNet broadcast proposals when compared in four well-known scenarios, as shown in the simulations carried out in Section 4.

2. Related work

In this section, we study the state of the art of data broadcasting in Opportunistic Networks (OppNet). We pay special attention to node characterisation in these networks by providing a summary of the different ways of studying node's centrality, similarity and storer reliability. Additionally, we present a summary of articles that use OST-based techniques to improve routing in OppNet.

2.1. Message broadcasting in Opportunistic Networks

Opportunistic Networking [14] is a research field that studies networks where end-to-end communication is not always guaranteed. OppNet nodes use intermediate nodes to route messages from their source to their destination in an opportunistic way.

Message broadcasting in this field studies the problem of node communication in a many-to-many, any-to-many, and one-to-any way. Traditional routing protocols like ProPHET [29], Bubble rap [24] or SPRINT [12] can not be directly applied to OppNet broadcasting because these proposals are defined for one-to-one (Unicast) communications. Sobin, et al. [36] present a complete survey of routing and data dissemination in Opportunistic Networks. Regarding message broadcasting, this survey presents a new taxonomy where data dissemination protocols are divided into two categories: social-based data dissemination schemes and pure opportunistic ones. The social-based schemes include publish/subscribe approaches such as the study by Yoneki, et al. [40], where communities are formed by interconnecting nodes that frequently meet with each other. Other non-publish/subscribe schemes, such as the work by Gao, et al. [19], propose different social metrics and utility-based data dissemination methods. Following the second category presented in [36], pure opportunistic data dissemination protocols can be either reactive, where the source node or nodes push data towards the destinations [44], or proactive where the subscribers pull the data from the nodes [21].

There are few OppNet broadcast schemes proposals that have taken into account the energy consumption. In [20], the authors propose a broadcast protocol where nodes wait for an opportunity to reach multiple nodes with one transmission in order to reduce the number of transmissions overall. Additionally, in [19], the authors propose an efficient dissemination protocol for OppNet that uses a social centrality metric to ensure effective relay selection. They show, by means of simulations, that their approach obtains better cost-effectiveness than existing data dissemination schemes.

2.2. Node characterisation in Opportunistic Networks

In OppNet literature, different node characterisation metrics have been defined to help with routing decisions. In this article, we focus on node's centrality, similarity and reliability.

In order to evaluate the importance of a node in a network, there are many centrality metrics the OppNet scientific community use. The three most important are degree, betweenness, and closeness centrality. Degree centrality [31] measures the number of direct edges that reach a given node. Betweenness centrality [7], calculates the number of shortest paths connecting other nodes that use the node being measured as a hop. Finally, closeness centrality [30] studies the length of the shortest path connecting the rest of the nodes. Several studies propose to use these metrics to assist with routing decision: nodes with high centrality are presumed to be good communication hubs.

Similarity or clustering has been studied in OppNet to detect nodes that belong in the same community. Routing decisions are made by comparing the similarity with the destination node. Examples of these routing protocols include classical studies like [22] and [13].

The ability of a node to be trusted in terms of how reliable it is to delegate a copy of a message has been studied from different perspectives. There is a large number of studies that propose different reputation-based schemes, such as [41] to address the problem of nodes not forwarding or accepting messages. These reputation schemes try to detect malicious nodes which are excluded from the routing process. Other proposals simply select nodes for message forwarding in terms of how reliable a node is according to network metrics such as how frequent a node drops its stored messages [35].

2.3. Optimal stopping theory in OppNet

In this article, we propose an efficient broadcasting protocol for OppNet messages. The efficiency of this protocol is achieved by limiting the number of message storers in the network. Every node contacted by the message creator node is studied as a candidate for storing and broadcasting the message, but only those that are good enough are selected. This local problem can be seen as an optimisation problem. Probabilistically speaking, early contacted nodes will not guarantee optimal results, that is, poor nodes will be selected for message storing. Late decisions, moreover, will also fail as it is likely that good candidates will have been discarded. OST [34] deals specifically with this type of problems. It is a statistical solution for the problem of choosing the best moment to make a particular decision to maximise a certain reward or to minimise a certain cost.

One of the most popular problems in OST is the secretary problem [17], also known as the Classical Secretary Problem (CSP). In this problem, a person must interview a group of n candidates, that can be ranked from best to worse, with the aim of selecting the best one. The difficulty of this problem lies in the fact that once a candidate is not selected, he/she cannot be recalled again. The solution to this problem, as presented in [17], is a selection strategy that discards the first n/e candidates interviewed (e being the mathematical constant) and selects the first next one that is better than all of the previous ones, if found.

From the CSP, more extensive Generalized Secretary Problems (GSP) are obtained if, instead of only the best possible candidate, a set of k good enough candidates have to be chosen. These k candidates can be the k best ones, or a combination which maximises a certain reward or minimises a certain cost. In particular, Tamaki proposes in Theorem 4 of [38] an optimal policy to find the first and the second best candidates in a GSP, and Ano proposes in Theorem 4 of [6] an optimal policy to find the first, second and third best candidates in a GSP.

In the context of opportunistic wireless networks, OST is a very powerful tool to help with different network decisions. In the field of distributed opportunistic scheduling [43], different proposals such as [33,42] employ OST techniques to optimally utilize wireless resources. The key of these studies lies in the fact that the wireless channel quality changes with time and therefore, OST plays an important role by choosing the best moment to perform certain network decisions. An example of these is finding the optimal data transmission duration time with the goal of minimizing energy consumption. [26] also uses OST to achieve significant energy cost reduction in the context of epidemic information dissemination in ad hoc networks, particularly in the route discovery phase of the advanced on-demand distance-vector (AODV) routing protocol.

Another OppNet proposal that uses OST is [8], in which the authors present Softwarecast, a general delivery scheme for group communications based on mobile code. In Softwarecast, messages carry a software code and a delivery state that permit them to perform refined delivery-decision-making methods based on OST to implement complex delivery decisions.

Table 1
Summary of notation.

Variable	Description
$cent(n_1)$	Node n_1 centrality.
$sim(n_1, n_2)$	Nodes n_1 and n_2 similarity.
$rel(n_1)$	Node n_1 reliability.
S	Maximum number of storers.
$s(n_1, n_2)$	Storer suitability metric.
TTL	Remaining life time of the message.
ict	Node inter-contact time.
$sict$	Moving average of ict.
$lastclist$	List of nodes recently contacted.
$maxlist$	kth best values of $s(n_1, n_2)$.
$\|C\|$	Number of nodes a node will contact before a message TTL has expired.
ns	Estimation of $\|C\|$ (node scope).

Moreover, in [23], the authors propose OSDR, an OppNet routing protocol that defines a utility function based on the average meeting time between an OppNet node and the destination of a message. Messages are replicated in terms of an OST strategy that achieves a minimum delivery message delay.

In [9], the authors present Relcast, a delivery paradigm where messages are sent to profiles defined by relative delivery functions such as *best, maximum* or *over-the-average*. Nodes belong to these relative profiles taking into account not only attributes from the very same node but also relative to others from the same profile. Additionally, they present Explore and Wait, a composite routing-delivery scheme that uses OST-based delivery strategies to route Relcast messages.

Finally, there are some very interesting proposals that use OST to find the optimal time to delay the reporting of information in the context of opportunistic mobile sensor networks [3]. In this context, the quality of the information gathered decreases over time. In order to optimise the quality of the context information delivered, proposals like [2–5] optimally decide on the moment to stop the process of gathering context information and send it. This optimality is achieved by proposing different time-optimized models based on OST.

As reviewed, literature has many entries on broadcast protocols for OppNet. However, the designing of these protocols does not stay within the basic criteria of efficiency, but mainly considers classical metrics like low latency or wide message dissemination. When it comes to efficiency, preventing unrestrained propagation of messages is crucial, and therefore storer nodes must be carefully selected. In this article, we want to explore a novel way of dissemination broadcast messages in OppNet that, contrary to all of the previous unicast and broadcast OppNet proposals presented in this section, does not limit the number of message replicas in the network, but the number of storers of the message. Nodes that receive the broadcast message, in turn, can not disseminate it again, unless they are chosen by the previous storer. This section introduced centrality, similarity and reliability of nodes as metrics that have been customarily used by routing algorithms to evaluate forwarding nodes. Unfortunately, merely using these metrics does not guarantee that messages are going to spread across the whole network (mostly because of the limited network capacity). OST provides a mechanism to hold back each storer selection until a favourable context is found. In this article, we use the metrics explored in this section to define an OST reward function to select good enough storers for an efficient broadcast, by optimally solving a different GSP in each of the stages of our protocol. In the next section, we fully present our proposal (Table 1).

3. Efficient broadcast

In computer networks, message broadcasting includes different network paradigms, such as one-to-many and many-to-many, where messages are broadcast without necessarily expecting a response from the recipients. In this section, we present a broadcast dissemination protocol that achieves high dissemination capacity, with a low number of excluded nodes, while remaining very efficient in terms of energy consumption.

Our proposal limits the number of storers in the network for a certain message, that is, the number of nodes that are allowed to forward the message. This broadcast dissemination protocol is explained in Section 3.1. In Section 3.2, we introduce a metric used by the presented protocol to characterise candidate nodes for the selection of the storers. This metric defines nodes' storer suitability in terms of node's centrality, custody reliability and contacted node similarity. Since it is difficult to know which are good or bad values for this metric, as this depends not on the node itself but on how higher or lower this metric value is relatively to other nodes', we propose in Section 3.3 a method to select nodes for the delegation of message storing. This node picking method is not trivial. The selection is divided in to several stages, each one using an independent OST based strategy using the previously introduced suitability metric. Finally, since this storer decision is performed in terms of how many nodes a node could potentially contact during the lifetime of a given message, we propose in Section 3.4 a means of estimating the number of nodes a certain node can potentially contact during a certain period of time.

At the end of Section 3, after a complete description of our broadcast dissemination protocol, we include Algorithm 1. The algorithm can be seen as a synthesis of the whole Section 3, and, to improve its readability, particular lines will be referenced throughout this section.

3.1. Routing protocol

Traditional network schemes include network primitives such as *routing* and *delivery*. *Routing* is the action of relaying messages from one node to another thus defining a communication path. When a message is routed from one node to another, the latter is allowed, in turn, to store the message in order to route it as well. On the flip side, the *delivery* action is defined as the action of allowing one or more applications in a node to receive the payload of the message. In our broadcast proposal, we define a strict upper bound on the number of nodes that can store broadcast messages. When an efficient broadcast message is created by a node (lines 15–19 of Algorithm 1), a particular number S is associated to that message, and the first stage of our protocol starts. Initially, S indicates the maximum allowable number of storers of the message in the network. The node that creates the message becomes the first storer of the message and it will replicate the message with different values of S. Each time a new storer is chosen and it gets a replica of the message, an independent new stage of the protocol starts (lines 32–33, 38–42 and 46 of Algorithm 1). Following this, we describe how a storer has to deal with a message it receives that is associated to a particular value of S:

- $S = 0$: the node is allowed to locally deliver the message but not to forward it to any other node.
- $S = 1$: the node can forward the message to as many nodes as it will contact. The S value in the new forwarded message is 0.
- $S > 1$: As in the previous case, the node can forward the message to as many nodes as it contacts. The S value in the new forwarded message is also set to 0. Additionally, it chooses a node to delegate the storing of the message, and forwards the message to it with a new S value of $\lceil S/2 \rceil$. The S value of the local copy is then set to $\lfloor S/2 \rfloor$ and the procedure is repeated considering the new value of S. Following the algorithm, a total number of $\lfloor log_2 S \rfloor$ nodes are selected as storers. The storers are selected using an OST based strategy in the context of this stage. To avoid repeating storers, each node keeps a list of all already forwarded messages and refuses to accept them again in the future. This mechanism is used in some epidemic proposals, such as [39].

Algorithm 1 Broadcast dissemination behaviour.

```
1: ▷ $msg$ is the broadcast message.
2: ▷ $msg.contacted$ is the number of nodes the node has contacted since the creation of the message.
3: ▷ $msg.maxlist$ are the $k$th best values for $s$.
4: ▷ $msg.p$ is the number of the storer to be delivered (first, second,..., $k$th).
5: ▷ $msg.k$ is the number of storers to select in the stage.
6: ▷ $msg.S$ is the maximum number of storers left.
7: ▷ $msg.TTL$ is the remaining life time of the message.
8: ▷ $localnode$ is the local node.
9: ▷ $contactednode$ is the contacted node.
10: ▷ $lastcontacttime$ is the last time the node was contacted.
11: ▷ $v_{ki}$ is the $i$th optimal stop value for the optimal selection of k stores (see Table 2).
12: ▷ $\alpha_i$ are the metric weights to compute s.
13: procedure NEWMESSAGE($msg$)
14:     #Procedure called when creating a message and starting the first stage.
15:     $msg.S$ = MAX_NUMBER_OF_STORERS
16:     $msg.contacted = 0$
17:     $msg.p = 1$
18:     $msg.maxlist = []$
19:     $msg.k = \lfloor log_2 msg.S \rfloor$
20: end procedure
21: procedure ONNEWCONTACT($contactednode, msg$)
22:     #Procedure called each time localnode contacts a new node, for every broadcast message.
23:     $msg.contacted++$
24:     $cent$ = centrality($contactednode$)
25:     $sim$ = similarity($localnode, contactednode$)
26:     $rel$ = reliability($contactednode$)
27:     $s = \alpha_1 \times cent + \alpha_2 \times sim + \alpha_3 \times rel$
28:     $ict$ = time.now() - $lastcontacttime$
29:     $sict = sict + \sigma \times (ict - sict)$
30:     $ns = msg.TTL/sict$
31:     $c = msg.contacted + ns$
32:     $replica$=$msg$.replicate()
33:     $replica.S=0$
34:     #Conditional statements described in Section 3.3:
35:     for $i = 1$, $i \le k$, $i$++ do
36:         #Check if contactednode has to be a new storer. If so, a new stage starts:
37:         if $(s \geq maxlist[i]$ & $msg.contacted > c \times v_{ki})$ $||(msg.contacted > (c - k + msg.p))$ then
38:             $replica.S = \lceil msg.S/2 \rceil$
39:             $replica.k = \lfloor log_2 replica.S \rfloor$
40:             $replica.resetvalues(maxlist, contacted, lastcontacttime, p)$
41:             $msg.S = \lfloor msg.S/2 \rfloor$
42:             $msg.p++$
43:             break
44:         end if
45:     end for
46:     $replica$.forward($contactednode$)
47:     #Update maxlist:
48:     $msg.maxlist$.updatemax($s$)
49: end procedure
```

For the sake of clarity, we provide in Fig. 1 an illustrative example where a message is sent from node n_1 using the efficient

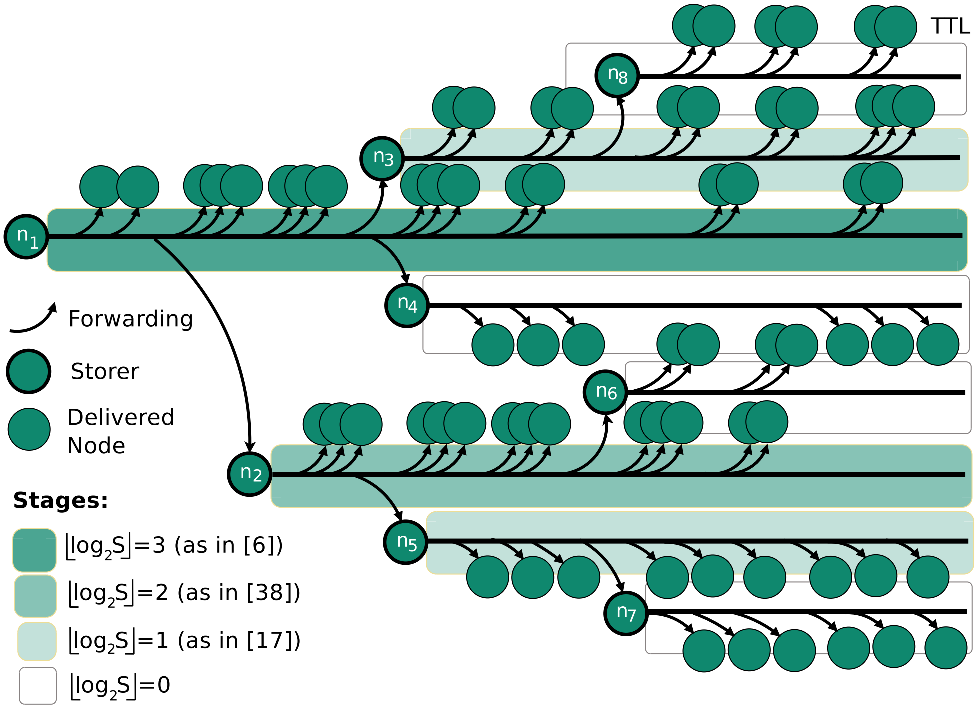


Fig. 1. Broadcast dissemination protocol strategy. Example with $S = 8$ storers.

broadcast proposed in this paper with an initial value of $S = 8$. The distinct stages of the protocol are depicted using horizontal stripes of different colours. In the first stage, following the algorithm, n_1 selects n_2, n_3, and n_4 as new storers, forwards the message to them using $S = 4$, $S = 2$ and $S = 1$ values respectively, and goes updating its own S value until a final value of $S = 1$. These storers are selected using a GSP strategy in the context of this first stage. While in the process of selecting the new storers, n_1 forwards the message to all nodes on sight using a value of $S = 0$ for each new copy. Next, in a new stage of the protocol, n_2 will repeat the procedure with the local copy of the message with $S = 4$. This time, n_5 and n_6 are selected as new stores following a totally separated GSP strategy, and copies of the message are forwarded to them with $S = 2$ and $S = 1$ values, respectively. At this point, n_5 and n_3 can still choose a new storer because they still have $S = 2$, and using independent CSP strategies in their stages they choose n_7 and n_8, and forward the message with $S = 1$.

By following this scheme, as it can be seen in Fig. 1, there are a total of eight storers of the message in the network (n_1 to n_8). The rest of the nodes in the figure have received a copy of the message with $S = 0$ for local delivery, and therefore they are not storers and cannot further forward the message.

As we will see in Section 3.3, each stage in the protocol will be related to a different GSP instance, each one with its own optimal solution.

3.2. Storer suitability

In this section, we propose a metric, $s(n_1, n_2)$, to compare a node's suitability to store a broadcast message. This metric is used in the reward function that will be used for storer selection, see Section 3.3. Using this metric, nodes are quantified in terms of three different characteristics:

- **Centrality**. We measure how well physically connected a node is. The more connected the node is, the more likely it will find more nodes to broadcast the message.
- **Similarity**. We study how similar a node is to the studied candidate for message storing delegation. The more dissimilar these nodes are, the more likely the candidate will find new and different nodes to broadcast the message.
- **Reliability**. We measure how reliable a node is for storing messages. The more reliable a node is, the more likely that it will disseminate the message.

For a given storer of a message n_s we define the suitability of a contacted node n_c as (lines 24–27 of Algorithm 1):

$$s(n_s, n_c) = \alpha_1 \times cent_{n_c} + \alpha_2 \times sim_{n_s,n_c} + \alpha_3 \times rel_{n_c} \qquad (1)$$

where $cent_{n_c}$ is how central the contacted node is, sim_{n_s,n_c} how similar the storer node is to the contacted node, rel_{n_c} how reliable for the storage of a message the contacted node is, and $\sum_{i=1}^{3} \alpha_i = 1$.

3.2.1. Centrality

Regarding the centrality metric, we calculate it for every node using its ego-centric network [18]. In this network, nodes' contacts are represented by an adjacency symmetric matrix where its rank, n, is the number of contacts a given node has made. For this purpose, every node keeps a contact matrix (CM) that represents the node's contact ego perspective of the network [18].

We propose to use the betweenness egocentric centrality as our centrality metric, a centrality metric with a small computational cost. As presented in [16], for a certain node n_i, this metric is calculated by computing the number of nodes that are indirectly connected through the ego node using the following expression:

$$cent_{n_i} = CM^2 \times [1 - CM]_{i,j} \qquad (2)$$

3.2.2. Similarity

For the similarity metric, we study how different a storer node is in comparison to the contacted candidate for message storing. Every time two nodes met, we compute a metric that calculates the intersection of the last contacted nodes from both nodes, i.e., when a given storer node n_s contacts a given node n_c, the similarity for node n_c is calculated in the following way:

$$sim_{n_s,n_c} = 1 - (|lastclist_{n_s} \cap lastclist_{n_c}|/|lastclist_{n_s}|) \tag{3}$$

being $lastclist_{n_i}$ node n_i's list of its last contacted nodes.

3.2.3. Reliability

For defining the reliability of a node, we consider two variables: *slastdrop* (moving average for the percentage of non-dropped messages from a certain period of time) and *freebufferspace* (free available buffer).

The *slastdrop* moving average is calculated every hour as:

$$slastdrop_{new} = slastdrop_{old} + \rho \times (lastdrop - slastdrop_{old}) \tag{4}$$

where ρ is the moving average weight factor.

For a certain node n_i, the reliability compound metric is calculated giving the same weight to both variables:

$$rel_{n_i} = slastdrop \times 0.5 + freebufferspace \times 0.5 \tag{5}$$

3.3. Storer selection

If a node n_s finds a node n_c with a certain value for the storer suitability $s(n_s, n_c)$, as defined in the reward function from the previous section, it is hard to know whether this value is a high one or a low one in comparison to what this node could find in future contacts. In this section, we analyse from an OST perspective, the strategy to select a set of good storers for a given message m with a maximum storer suitability value for $s(n_s, n_c)$.

Being *TTL* the remaining life time of the message and n_i a certain node, we define the contact node list $C(n_i, TTL)$, as the list of future contacts for node n_i for a period of time *TTL*:

$$C(n_i, TTL) = \{c_1, c_2, ..., c_{|C|}\} \tag{6}$$

Note that the cardinality of C is normally unknown in OppNet. This means that a given node n_i cannot necessarily know in advance the number of nodes it will contact before the message m expires. In Section 3.4, a method for estimating this cardinality will be presented. For example, in Fig. 1, node n_1, will find 20 nodes before the broadcast message will expire.

As presented in the previous section, given a node n_c, candidate for message storing, and n_s, the current storer of the message, we define $s(n_s, n_c)$ as a way of evaluating node n_c's storer suitability.

The future contacts for a given storer n_s in a network, in a hypothetical situation when seen together, may be ordered from best to worst unambiguously in terms of $s(n_s, n_c)$. Moreover, we define the criterion of order of a set of nodes $P \subseteq C$ as:

$$t_P = \sum_{m \in P} s(n_s, n_m) \tag{7}$$

Eq. (7) allows us to define how good the composite choice of a set of candidate nodes is for storing delegation. For example, in Fig. 1, for the storer n_1, we calculate $t_{\{n_2,n_3,n_4\}}$ to evaluate how good the choice of the set $\{n_2, n_3, n_4\}$ is for storing delegation.

Additionally, we define $\mathcal{P}$ as the set of all of the possible subsets of C:

$$\mathcal{P} = \{\{n_{i_1}, n_{i_2}, ..., n_{i_r}\} \subseteq C \setminus \emptyset\} \tag{8}$$

Additionally, let $\mathcal{P}_k \subseteq \mathcal{P}$ be the set containing all the subset of $\mathcal{P}$ for a certain size k:

$$\mathcal{P}_k = \{P \in \mathcal{P} |\ |P| = k\} \tag{9}$$

As explained in Section 3.1, for a given message m, a node n_i contacts different nodes until it delegates storing for that message m to $k = \lfloor log_2S \rfloor$ nodes from the set C using opportunistic contacts. Let $n_{i_{t+1}}$ be a new forwarded node for potential storing delegation. The delegation condition, f, is used to decide whether storing of this message should be delegated or not to $n_{i_{t+1}}$ based only on the node and the previously forwarded nodes:

$$f : CxP \rightarrow \{0, 1\} \tag{10}$$

We want to delegate storing of a message m to $\lfloor log_2S \rfloor$ and only $\lfloor log_2S \rfloor$ nodes $P = \{n_{i_1}, n_{i_2}, \cdots, n_{i_{\lfloor log_2S \rfloor}}\} \in \mathcal{P}_k$ that maximises t_P. A solution to the problem has to define a strategy to maximise the probability of selecting a set in $\mathcal{P}_{\lfloor log_2S \rfloor}$, $\mathcal{O}$, such that:

$$\mathcal{O} = \{P \in \mathcal{P}_{\lfloor log_2S \rfloor} \mid t_P \geq t_{P'}\ \forall P' \in \mathcal{P}_{\lfloor log_2S \rfloor}\} \tag{11}$$

This is equivalent to maximise a reward function $\mathcal{W}$ that returns 1 if the $\lfloor log_2S \rfloor$ nodes that fit best to the targeted criteria are selected and 0 otherwise. Given a list of nodes $P = \{n_{i_1}, n_{i_2}, \cdots, n_{i_{\lfloor log_2S \rfloor}}\} \in \mathcal{P}_{\lfloor log_2S \rfloor}$ selected as new storers, we define $\mathcal{W}(P)$:

$$\mathcal{W}(P) = \begin{cases} 1 & P \in \mathcal{O} \\ 0 & otherwise \end{cases} \tag{12}$$

For each stage of our protocol, this corresponds to an OST problem where the decision is whether a set of nodes should or should not be chosen as new storers when these nodes are contacted. Note that if the first $\lfloor log_2S \rfloor$ encountered nodes are chosen as storers it is quite likely that these nodes will not be the $\lfloor log_2S \rfloor$ best nodes in terms of $s(n_s, n_c)$. In the same way, if we wait for the last $\lfloor log_2S \rfloor$ contacted nodes, it will be equally likely that the best suitable nodes will be discarded. Furthermore, the method of deciding after encountering every node in the network is not feasible: nodes may never be re-encountered again.

This optimisation problem corresponds to the CSP [17] when $k = \lfloor log_2S \rfloor = 1$, to the GSP of [38] (Theorems 1 and 4) when $k = \lfloor log_2S \rfloor = 2$, and to the GSP of [6] (Theorems 1 and 4) when $k = \lfloor log_2S \rfloor = 3$.

We follow the methodology for the GSPs presented by Ano in [6], Theorems 1 and 4. This methodology, for the particular case of $k = \lfloor log_2S \rfloor$, is a strategy where there exists an optimal set of stopping values $\nu_1|C|, \nu_1|C|, \nu_2|C|, ..., \nu_{\lfloor log_2S \rfloor}|C|$, $1 \leq \nu_1|C| \leq \nu_2|C| \leq \cdots \leq \nu_{\lfloor log_2S \rfloor}|C| \leq |C|$ such that the pth storer for message m ($p \leq \lfloor log_2S \rfloor$) is:

- the first node which is the best of all the previously seen nodes, if the node has contacted $v_1|C|$ nodes.
- or, if not found, the first node which is the second best of all the previous nodes, if the node has contacted $v_2|C|$ nodes.

$\vdots$

- or, if not found, the first node which is the $\lfloor log_2S \rfloor$th best of all the previous nodes, if the node has contacted $\nu_{\lfloor log_2S \rfloor}|C|$ nodes.
- or, if not found, the $(|C| - \lfloor log_2S \rfloor + p)$th node.

The list of conditional statements enumerated above is represented in Algorithm 1 on lines 35 and 37, comparing the storer suitability of the contacted node with the best k values of the storer suitability of past contacted nodes, which are stored in the *maxlist* array. This array is updated in Algorithm 1 in line 48. This strategy, applied independently at every stage of the protocol, maximises $\mathcal{W}(P)$ from a statistical perspective, that is, maximises the suitability of the selected nodes for storing the broadcast message. Note that, even though the selection process may result in optimality in each stage of the protocol, we can not guarantee the global optimality of this selection, as the storers are not selected by a single node but by a set of distributed nodes, each of which follows an independent GSP in its stage of the protocol.

In Table 2, a list of the optimal stopping values for every statistical problem is presented along with the references used. Note, that this routing decision is in terms of the cardinality of the contact node list ($|C|$). In the following section, we explain how this cardinality can be estimated.

Table 2
Optimal stop values for the statistical problem of finding k storers, k values in $1-3$.

k	v_1	v_2	v_3	Reference
1	$1/e\vert C\vert$	–	–	[17]
2	$0.2291\vert C\vert$	$0.6065\vert C\vert$	–	[38]
3	$0.1668\vert C\vert$	$0.437\vert C\vert$	$e^{-1/3}\vert C\vert$	[6]

3.4. Node scope

As introduced in the previous section, the cardinality of the contact node list $|C|$ is normally unknown in OppNet because of the idiosyncrasy of the network: it is difficult to know in advance the number of nodes a node will contact before the expiration time of a message. In order to calculate an estimation of the number of nodes a certain node can contact before the expiration time of a message (*TTL*), a statistic of the time between contacted nodes is kept within the node. We call this statistic the smooth inter-contact time (*sict*). To prevent recently forwarded nodes distorting this estimation, nodes keep a list of the last nodes the node has contacted. We call this list *last contacted node list (lastclist)*.

Every time a node contacts another node not included in the list *lastclist*, the observed inter-contact time (*ict*) is calculated (line 28 of Algorithm 1):

$$ict = time.now() - lastcontacttime, \tag{13}$$

where *time.now* is the current time and *lastcontacttime* is the last time the node contacted another node.

The *sict* statistic is updated using an EWMA (Exponential Weighted Mobile Average) in the following way (line 29 of Algorithm 1):

$$sict_{new} = sict_{old} + \sigma \times (ict - sict_{old}), \sigma \in [0, 1], \tag{14}$$

where $sict_{old}$ is the historical *ict*, σ is a weight factor and *ict* the last *ict* measured.

The estimated the number of nodes a certain node can contact before a broadcast message expires $ns(TTL)$ (node scope) is calculated using the following expression (line 30 of Algorithm 1):

$$ns(TTL) = TTL/sict. \tag{15}$$

The estimation of $|C|$ for a certain node n_i for the proposed optimal routing strategy proposed in Section 3.3 can be calculated as (line 31 of Algorithm 1):

$$|C| \widehat{=} \; contacted_{n_i} + ns(TTL), \tag{16}$$

where $contacted_{n_i}$ is the number of contacted nodes by node n_i.

In this section, we presented our efficient broadcast dissemination protocol for OppNet. In the next section, we will evaluate its performance and compare our approach to others from the state of art presented in Section 2.

4. Evaluation

In this section, we present the experimentation carried out to evaluate the performance of our proposed broadcast dissemination protocol. First, we state the goal of the evaluation and the metrics we use. Then, we describe the simulation environment, network scenarios and simulation parameters. Finally, we present the achieved results.

4.1. Goal and performance metrics

The goal of the experimentation is not only to evaluate our proposed protocol but also to compare its performance with that of other protocols representative of the state of the art regarding message broadcasting in OppNet.

First of all, this analysis and comparison are achieved by using three classical metrics:

- **Broadcast message dissemination:** number of nodes a broadcast message arrives at.
- **Energy consumption:** energy needed to broadcast a message.
- **Message latency:** average time for broadcast message delivery.

Second, we use these three metrics in a compound manner and analyse the broadcast efficiency, *i.e.* we study message dissemination per unit of energy consumption and message latency.

Third, we analyse the number of dropped messages to check the use of the network storage capacity.

Last, we analyse the number of excluded nodes, i.e. the number of nodes which do not receive any broadcast message.

4.2. Simulation environment

The experiment has been performed using the Opportunistic Network Environment (TheONE) simulator [25]. We have enhanced this simulator to include our proposed broadcasting protocol and for comparison purposes, other OppNet broadcast proposals from the state of the art.

We choose four different scenarios to analyse the performance of our proposal. Node contacts from the four scenarios are defined by physical contacts obtained from real mobility traces from the Crawdad database[1], a community resource for collecting wireless data at Dartmouth College, United States.

The first scenario, the *Info5* scenario, is based on real mobility traces, as published in [1]. These traces were retrieved during the 2005 edition of the Infocom conference in the course of 2.97 days. Contacts from these mobility traces represent 41 students. The total number of physical encounters provided in these traces is 22459.

The second scenario, the *Cambridge* scenario, is based on real contact traces from 51 students from the System Research Group of the University of Cambridge carrying small devices for six days [27]. Additionally, some stationary nodes were placed at various points of interest such as grocery stores, pubs, market places, and shopping centres all around the city of Cambridge, UK. A stationary device was also placed at the reception of the Computer Lab, in which most of the experiment participants were students. The number of contacts provided in these traces is 10641.

The third scenario, the *Taxis* scenario, as published in [10], contains mobility traces from 370 taxi cabs in Rome over 30 days. The number of contacts provided in these traces is 449226.

The fourth scenario, the *MIT* scenario, as published in [15], represents the activity information from 97 subjects at Massachusetts Institute of Technology over the course of the 2004–2005 academic year (196 days). The number of contacts provided in these traces is 205187.

In Fig. 2, we analyse the total contacts and unique contacts for every node for the different scenarios. As it can be seen, the Cambridge and MIT scenarios have a low number of contacts while in Info5 and Taxis the number of contacts is high. We see that in the Info5 scenario, the ratio between the total number of contacts and the unique contacts is very large in comparison to the other three scenarios.

In Fig. 3, we depict the contact social network for the Info5, Cambridge, Taxis and MIT scenarios. In Table 3, a scenario characterisation in terms of classical network topology measures is presented.

[1] http://crawdad.org/keyword-DTN.html.

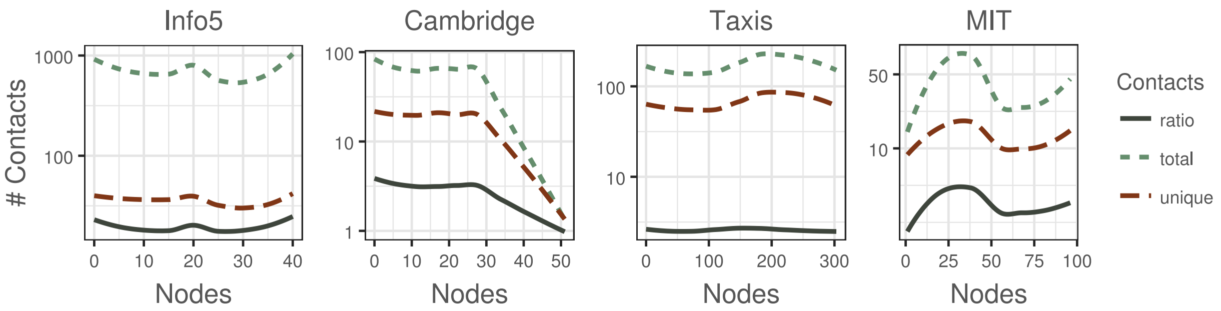
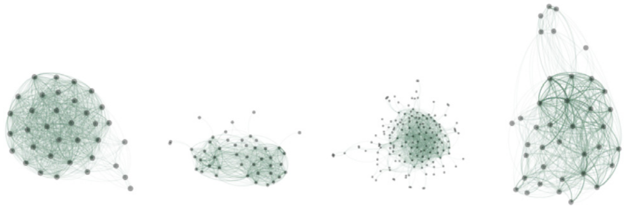


Fig. 2. Total and unique contacts for every node for the different studied scenarios.



(a) Info5. (b) Cambridge. (c) Taxis. (d) MIT.

Fig. 3. Contact social network for Info5, Cambridge, Taxis and MIT scenarios.

Table 3
Scenario characterisation.

Variable	Info5	Cambridge	Taxis	MIT
Density	High	Low	Low	High
Modularity	Low	Medium	Low	High
Clustering Coeff.	High	High	Medium	High
Communities	6	3	4	4
Average Path length	Low	High	High	Medium
Presence of hubs	No	Yes	Yes	No

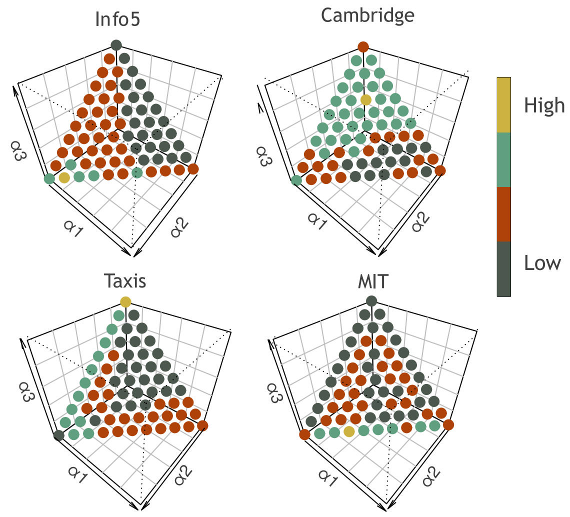


Fig. 4. Broadcast efficiency as a function of the different values for α_1 (centrality weight), α_2 (similarity weight) and α_3 (reliability weight).

As can be seen from Figs. 3 and 2, and Table 3, the four scenarios are different in terms of their contact network topology.

4.3. *Protocol parameters for simulation*

The simulations require particular values for certain parameters in order to get useful results. Specifically, the protocol needs specific values for the weights of centrality, similarity and reliability in the node suitability metric, as described in Section 3.2, as well as the energy cost of basic operations.

4.3.1. *Metric weights*

The weights of centrality, similarity and reliability in the node suitability metric (namely α_1, α_2, and α_3, where $\sum_{i=1}^{3}\alpha_i = 1$), determine the behaviour of the protocol. The best values for these parameters may depend on the specific scenario where the protocol is in operation, but for the sake of comparison, simulations of different scenarios should share these weights. In order to choose values that do not favour a particular scenario, we have analysed the outcome of the suitability metric for different combinations of values in all scenarios. Fig. 4 shows the efficiency performance of a broadcast message for the different scenarios and the different values of α_1, α_2 and α_3. As it can be seen, for the scenarios Info5 and MIT, low values for α_3 obtain a higher efficiency performance. This is probably due to the fact that in both scenarios there is a high density of contacts, and reliability is not really a decisive factor. Instead, for the Cambridge and Taxis scenario, the best results seem to come with lower values of α_1, probably due to the fact that in these scenarios there are hotspots that facilitate contacts and minimise the relevance of the intrinsic centrality of nodes.

For the simulation, a safe triplet of values for the α_i constants is chosen, providing no significant differences between scenarios, which provides a perfect balance of the three weighted metrics:

$$\alpha_1 = \alpha_2 = \alpha_3 = \frac{1}{3}$$

Fig. 4 shows that this combination does not favour any particular scenario.

4.3.2. *Energy consumption*

When calculating efficiency, it is crucial to determine the amount of energy consumed. To introduce an accurate estimation of energy consumption in the simulation, we conducted a practical experiment which measures real consumptions of data communications.

In the experiment, we used two Raspberry Pi devices with WiPi WiFi dongles, powered by AC 230 V - DC 5 V transformers. The electrical feed was monitored by two USB ammeters, as shown in Fig. 5. One of the computers was used to calculate the “idle” state energy consumption, while the second was performing the tests. The difference between the measured values provided the energy consumption of the test. To improve accuracy, each experiment was run continuously for two hours and repeated four times. The data from the first 20 minutes has been discarded to make sure the system was in a steady condition after booting. For each

Fig. 5. Energy consumption measurement: ammeters and Raspberry Pi.

Table 4
Energy consumption.

Operation	Average	Units
Determinants	5,332 × 10^{-3}	Ws/Mop (Joules/Mop)
Polynomials	1,20144 × 10^{-4}	Ws/Mop (Joules/Mop)
10K messages	151875 × 10^{-4}	Ws (Joules/message)

experiment, the considered result is the average of the median values of the four tests.

The aim of the experiments is to calculate the energy consumed by the routing algorithm. In Algorithm 1 there are three basic operations: sending of messages, matrix determinants, and polynomial equations. The tests regarding message sending consumption consisted of sending messages with a 10KByte payload. As for the operation tests, we used randomly generated matrices to calculate determinants and second grade polynomial calculations.

As seen in Table 4, determinants and polynomials energy consumption is negligible, as the routing algorithms use only a few of these operations (units in the table are in Joules per Mega operations). Each sent message consumed approximately 1.52×10^{-4} J.

4.4. Routing strategy

In Fig. 6, we study the impact of the value of the number of storers per message (S) on our protocol. We analyse three classical metrics: energy consumption, broadcast message dissemination and average latency time. As it can be seen, S has different impacts on the four studied scenarios. For most of the scenarios, the results for all three metrics show a substantial improve for $0 < S <= 6$. Instead, for $6 < S < 12$, most of the metrics stabilise or show a slight improvement. Some metrics start deteriorating when $S > 12$. $S = 12$ seems to be the best choice for our protocol in every scenario.

For the experimentation, as we want to compare our broadcast dissemination protocol to other three protocols representative of the state of the art, for the sake of fairness, we do not choose the best possible instance for our protocol ($S = 12$) but we study a generic *middle-case* where the maximum number of storers in the network is $S = 8$. With this choice, we also reproduce the example described in Fig. 1. See Section 3.1 for a complete description of the behaviour of each storer in this situation.

As a consequence of having an initial S value set to 8, we need to solve three different statistical problems, as introduced in Section 3.3: the GSP from [6] ($\lfloor log_2 S \rfloor = 3$), the GSP from [38] ($\lfloor log_2 S \rfloor = 2$) and the CSP [17] ($\lfloor log_2 S \rfloor = 1$). For every statistical problem the value for $|C|$ is calculated following the estimation from Eq. (16) (see Section 3.4).

4.5. Results

In order to study the performance of our proposal, we conduct several simulations where 400 broadcast messages are sent to the networks defined in the four different scenarios. In order to make these scenarios realistic, besides the studied broadcast messages, we are also including 1000 unicast messages that compete for the same network resources.

We compare our broadcast dissemination protocol (*"effi"* in the figures) with three broadcast dissemination protocols representative of the state of the art. On the one hand, we study the performance of a scheme where messages are epidemically forwarded to every contacted node in the network (*"epid"* in the figures) as in [28]. Then, we evaluate the performance of a broadcast dissemination protocol where central nodes in the network are selected to disseminate the broadcast messages (*"cent"* in the figures), as in proposals such as [19,32,37]. Finally, a third proposal is evaluated where broadcast messages are geography-limited by means of their message time to live mandating a restriction on the propagation of the messages within a specified number of hops or distance from the message source, as in proposals like [11] (*"copylim"* in the figures).

In Fig. 7(a)–(c) we analyse the performance of our proposal for these broadcast messages using the classical metrics: broadcast message dissemination, energy consumption and average latency delivery time, respectively. As it can be seen, our proposal performs well when compared to the other three in terms of message dissemination. Since our proposal efficiently limits the flooding of the message, due to the message dropping, our proposal performs in some scenarios (Cambridge and MIT) even better than the epidemic one. Concerning energy consumption, our proposal improves on the other three in a very significant manner for the four sce-

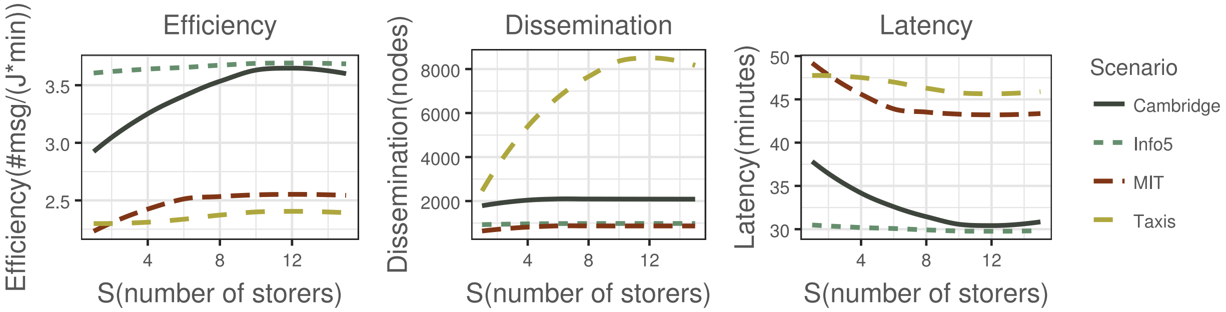
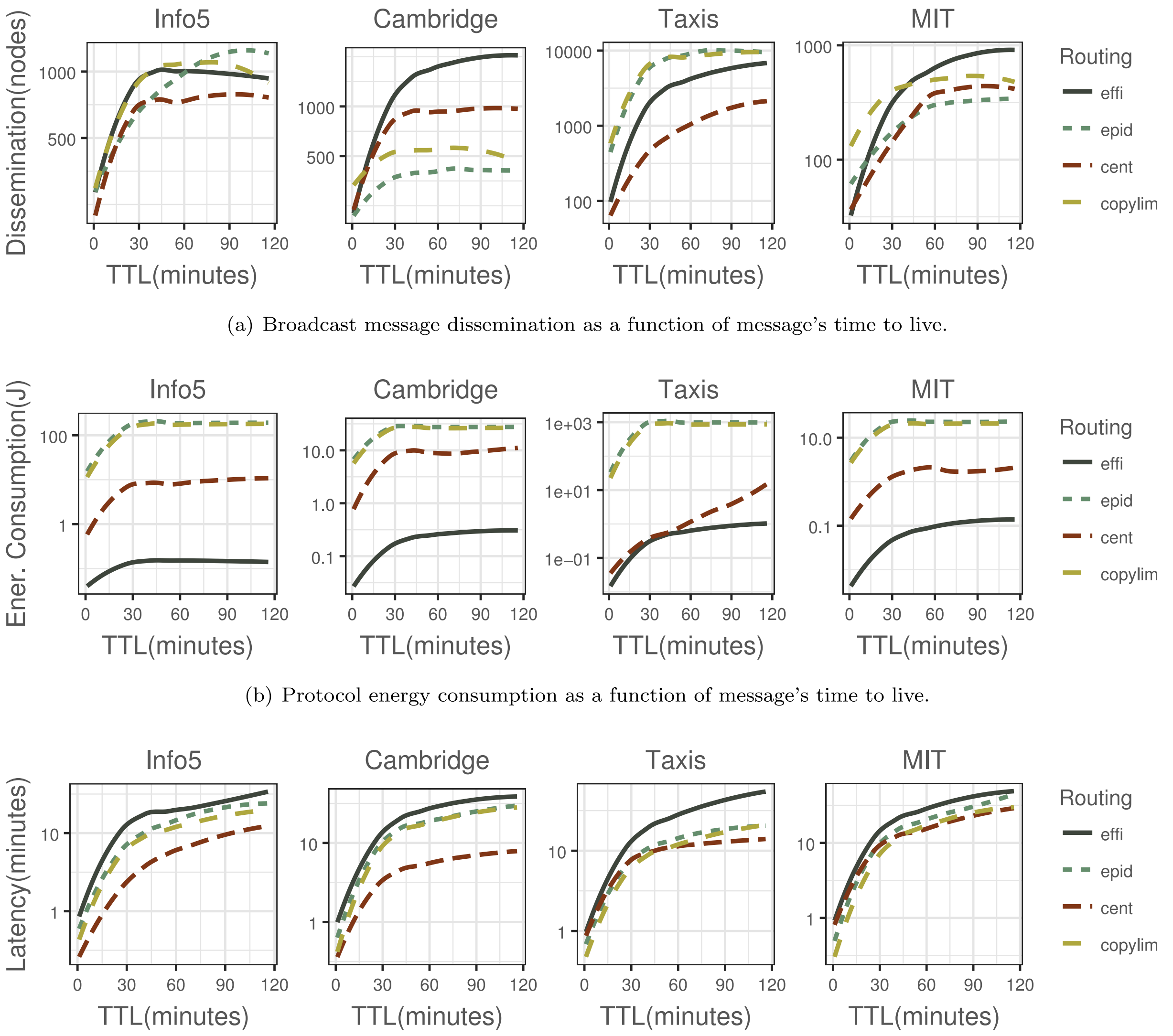


Fig. 6. Broadcast message dissemination, protocol energy consumption and average latency time as a function of the number of storers (S).



(a) Broadcast message dissemination as a function of message's time to live.

(b) Protocol energy consumption as a function of message's time to live.

(c) Average latency time as a function of message's time to live.

Fig. 7. Protocol performance: broadcast message dissemination, energy consumption and average latency time.

narios. Unfortunately, when it comes to message latency time, our proposal does not provide good results in comparison to the other three proposals, but this was somewhat expected, since we delay the selection of the storers.

If we consider these three metrics in a compound fashion, we can study the broadcast efficiency, that is, the message dissemination per unit of energy consumption and time to deliver the message. We analyse this efficiency in Fig. 8(a), as a function of the message time to live. As can be seen, in all scenarios, our proposal is a significant improvement on the state-of-the-art.

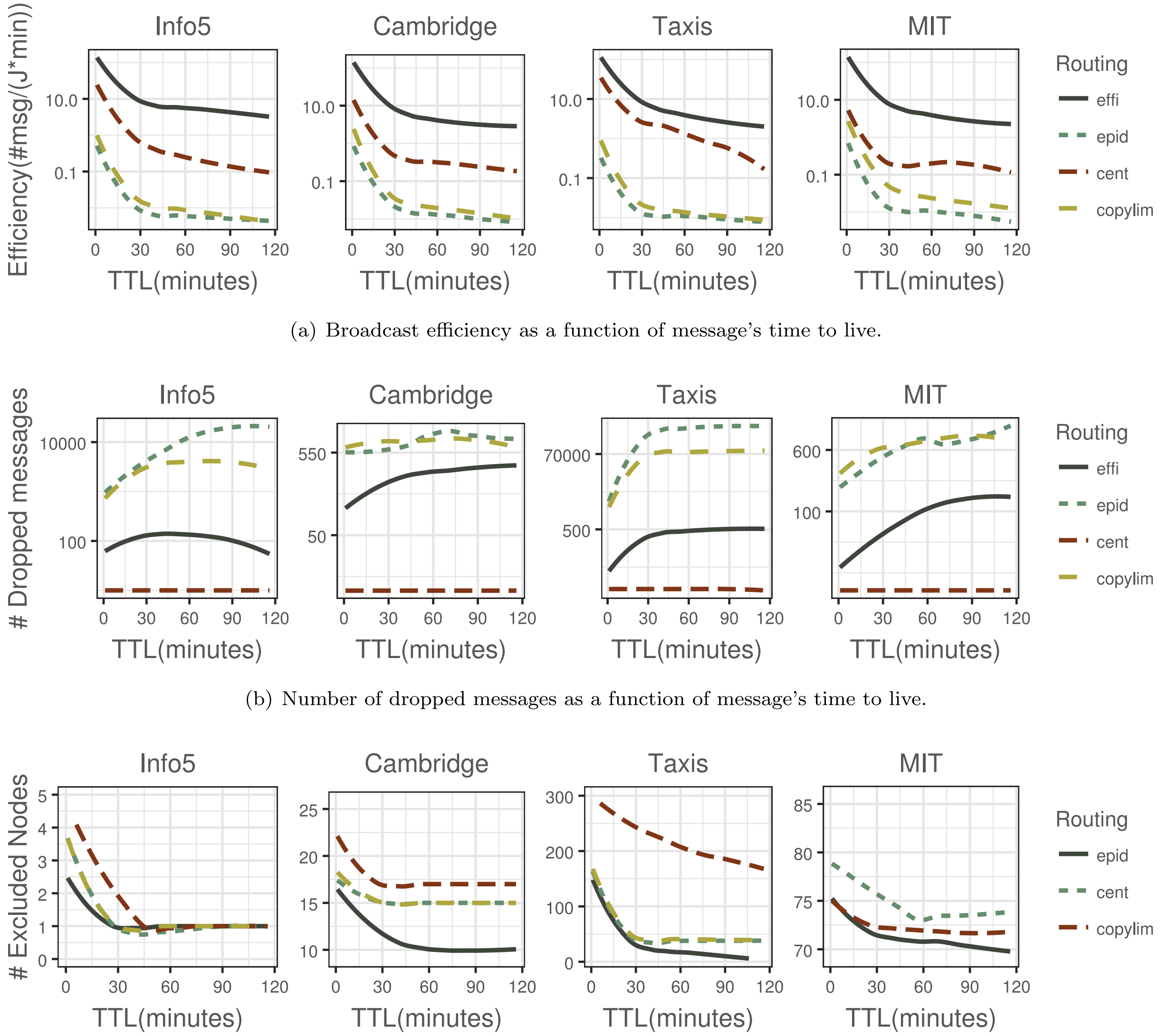
In Fig. 8(b), we analyse the number of dropped messages as a function of the message time to live. Our proposal greatly reduces this number, and thus improves the network storage capacity, in comparison with the *"epid"* and *"copylim"* proposals. Our proposed protocol cannot compete with the reduction offered by *"cent"* because this last proposal is very restrictive regarding the replication of any message sent. However, this very restriction leads *"cent"* to achieve poorer dissemination results than our proposal, as we can see in Fig. 7(a).

Finally, we analyse the distribution of the frequency of the number of nodes contacted by the 400 broadcast messages, paying attention to those nodes that are not contacted by any broadcast message (excluded nodes). In Fig. 8(c), we analyse the number of excluded nodes as a function of the message time to live. As it can be seen, our proposal matches the number of excluded nodes in comparison to the other broadcast protocols in the Info5 scenario, and reduces this number in the other three cases.

5. Conclusions

Most applications running on OppNets require broadcast delivery of messages (and other forms of multicast) for their normal operation, and having an efficient broadcast routing strategy for these messages is essential for the sake of real applications.

As opposed to other proposals devoted to the problem of message broadcasting in OppNet, which are based on limiting the number of message copies or their life span, in this paper we proposed an efficient (in terms of energy consumption and network capacity usage) broadcast dissemination protocol which uses a different perspective. Broadcast messages are limited not just regarding the number of copies or their life span, but also carefully selecting their prospective forwarders (storers).



(a) Broadcast efficiency as a function of message's time to live.

(b) Number of dropped messages as a function of message's time to live.

(c) Number of excluded nodes as a function of message's time to live.

Fig. 8. Routing performance: broadcast efficiency, number of dropped messages and number of excluded nodes.

In each stage of our proposed routing protocol we use a different GSP as the key element to devise a strategy which maximises, from a statistical perspective, the suitability of the list of the selected nodes for storing every broadcast message. Our strategy selects the best message storers in every stage, holding back broad message dissemination until convenient conditions are met. This strategy makes the most of the network storage capacity, facilitating and expediting the spreading of messages throughout the network. The metrics used to assess favourable forwarding conditions include betweenness egocentric centrality; node reliability measured as both the least dropping probability and the most space availability for messages; and the similarity among contacted nodes.

We evaluate our protocol using four well-known scenarios based on real mobility traces from real communication datasets, and comparing our protocol with three other broadcast protocols representative of the state of the art in OppNet broadcasting. We first analyse three classical performance metrics: broadcast message dissemination, energy consumption and message latency. Our proposal, when compared to the other proposals, performs well in terms of message dissemination; improves on the other three in a very significant way regarding energy consumption; with only a minor increase in terms of message latency. Second, we use these three metrics in a combined manner and analyse the broadcast efficiency, i.e. the message dissemination per unit of energy and message latency. In this regard, our work is a significant improvement beyond the existing protocols. Next, we analyse the number of dropped messages to evaluate the usage of network capacity. Our proposal greatly reduces the number of dropped messages, when compared with epidemic and limited-epidemic proposals, and is insufficient when compared to protocols which select central nodes to disseminate the broadcast messages. However, this limitation gets compensated by the higher dissemination results attained by our proposal. Finally, we analyse the number of excluded nodes, i.e. the number of nodes which are not contacted by any broadcast message. In three of the four scenarios, our work reduces the number of excluded nodes in comparison to the other broadcast protocols, while in the fourth scenario we mandate parity (Table 5).

Table 5
Simulation variable summary.

General	Value
Simulation duration	55 h
Transmission Speed	250 KB/s
# Nodes (Info5)	41
# Nodes (Cambridge)	51
# Nodes (MIT)	97
# Nodes (Taxis)	370
# random seeds	50
Buffer Size	100KB–10MB
Traces information	
# Contacts (Info5)	22459
# Contacts (Cambridge)	10641
# Contacts (Taxis)	449226
# Contacts (MIT)	205187
Message information	
# Broadcast Messages	400
# Unicast Messages	1000
Message size	10KB
TTL	5’–120’
Routing/Delivery Settings	
S	8
α_1, α_2, α_3	1/3
γ (ageing constant)	0.98
k (time unit ageing constant)	30 s
σ (*sict* moving average factor)	0.75
ρ (*slastdrop* moving average factor)	0.75
slastdrop time period	1 h

Acknowledgements

This work was partly supported by the Catalan AGAUR 2017SGR-463 project and the Spanish Ministry of Science and Innovation TIN2017-87211-R project. The authors thank Marc Dalmau for his generous help with the energy consumption experimentation. We also thank Dr. Jason Quinlan for his assistance reviewing the grammar of our manuscript.

References

[1] A. Dimitrios-Georgios, CRAWDAD dataset uoi/haggle (v. 2016-08-28): derived from cambridge/haggle (v. 2009-05-29), 2016. Downloaded from http://crawdad.org/uoi/haggle/20160828/one, August.

[2] A. Christos, Time-optimized contextual information forwarding in mobile sensor networks, J. Parallel Distrib. Comput. 74 (5) (2014) 2317–2332.

[3] C. Anagnostopoulos, H. Stathes, Delay-tolerant delivery of quality information in ad hoc networks, J. Parallel Distrib. Comput. 71 (7) (2011) 974–987.

[4] C. Anagnostopoulos, H. Stathes, Multivariate context collection in mobile sensor networks, Comput. Netw. 57 (6) (2013) 1394–1407.

[5] C. Anagnostopoulos, S. Hadjiefthymiades, Z. Evangelos, Optimal stopping of the context collection process in mobile sensor networks, in: In Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on, IEEE, 2013, pp. 2067–2071.

[6] A. Katsunori, Optimal selection problem with three stops, J. Oper. Res. Soc. Japan 32 (1989) 491–504.

[7] B. Marc, Betweenness centrality in large complex networks, Eur. Phys. J. B–Cond. Matter Complex Syst. 38 (2) (2004) 163–168.

[8] C. Borrego, G. Garcia, R. Sergi, Softwarecast: a code-based delivery manycast scheme in heterogeneous and opportunistic ad hoc networks, Ad Hoc Netw. 55 (2017) 72–86.

[9] C. Borrego, A. Sánchez-Carmona, Z. Li, R. Sergi, Explore and wait: a composite routing-delivery scheme for relative profile-casting in opportunistic networks, Comput. Netw. 123 (2017) 51–63.

[10] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, R. Antonello, CRAWDAD dataset roma/taxi (v. 2014-07-17), 2014. Downloaded from https://crawdad.org/roma/taxi/20140717, July.

[11] Q. Chen, S.S. Kanhere, H. Mahbub, Performance analysis of geography-limited broadcasting in multihop wireless networks, Wireless Commun. Mob. Comput. 13 (15) (2013) 1406–1421.

[12] R.I. Ciobanu, C. Dobre, C. Valentin, Sprint: social prediction-based opportunistic routing, InWorld of Wireless, Mobile and Multimedia Networks (WoWMoM),2013 IEEE 14th International Symposium and Workshops on a, pages 1–7., IEEE, 2013.

[13] E.M. Daly, H. Mads, Social network analysis for routing in disconnected delay–tolerant manets, in: In Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing, ACM, 2007, pp. 32–40.

[14] K.D. Mieso, Mobile opportunistic networks: architectures, Protocols and Applications, CRC Press, 2016.

[15] N. Eagle, A.S. Pentland, v..-.-. CRAWDAD dataset mit/reality, D.f. u. July, 2005.

[16] M. Everett, P.B. Stephen, Ego network betweenness, Soc. Netw. 27 (1) (2005) 31–38.

[17] S.F. Thomas, Who solved the secretary problem? Stat. Sci. (1989) 282–289.

[18] C.F. Linton, Centered graphs and the structure of ego networks, Math. Soc. Sci. 3 (3) (1982) 291–304.

[19] W. Gao, C. Guohong, User-centric data dissemination in disruption tolerant networks, in: In INFOCOM, 2011 Proceedings IEEE, pages 3119–3127., IEEE, 2011.

[20] A. Goundan, E. Coe, R. Cauligi, Efficient broadcasting in delay tolerant networks, in: In Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE, pages 1–5., IEEE, 2008.

[21] O. Helgason, K. Gunnar, Podnet: a system architecture for opportunistic content distribution, R. Inst. Technol. (KTH) (2010).

[22] T. Hossmann, T. Spyropoulos, L. Franck, Know thy neighbor: towards optimal mapping of contacts to social graphs for dtn routing, in: In INFOCOM, 2010 Proceedings IEEE, pages 1–9., IEEE, 2010.

[23] D. Huang, S. Zhang, C. Zhou, An optimal stopping decision method for routing in opportunistic networks, in: In Wireless Communications and Networking Conference (WCNC), 2013 IEEE, pages 2074–2079., IEEE, 2013.

[24] P. Hui, J. Crowcroft, Y. Eiko, Bubble rap: social-based forwarding in delay-tolerant networks, IEEE Trans. Mob. Comput. 10 (11) (2011) 1576–1589.

[25] A. Keränen, J. Ott, K. Teemu, The ONE simulator for DTN protocol evaluation, in: In Proceedings of the 2nd international conference on simulation tools and techniques, page 55. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[26] T. Kontos, C. Anagnostopoulos, E. Zervas, S. Hadjiefthymiades., Adaptive epidemic dissemination as a finite-horizon optimal stopping problem, Wireless Netw. (2018) 1–18.

[27] J. Leguay, P. Hui, J. Crowcroft, J. Scott, A. Lindgren, T. Friedman, CRAWDAD dataset upmc/content (v. 2006-11-17), 2006. Downloaded from http://crawdad.org/upmc/content/20061117, November.

[28] Y. Li, P. Hui, D. Jin, L. Su, Z. Lieguang, Evaluating the impact of social selfishness on the epidemic routing in delay tolerant networks, IEEE Commun. Lett. 14 (11) (2010) 1026–1028.

[29] A. Lindgren, A. Doria, S. Olov, Probabilistic routing in intermittently connected networks, ACM SIGMOBILE Mob. Comput. Commun.Rev. 7 (3) (2003) 19–20.

[30] K. Okamoto, W. Chen, L. Xiang-Yang, Ranking of closeness centrality for large-scale social networks, In International Workshop on Frontiers in Algorithmics, pages 186–195., Springer, 2008.

[31] T. Opsahl, F. Agneessens, S. John, Node centrality in weighted networks: generalizing degree and shortest paths, Social Netw. 32 (3) (2010) 245–251.

[32] J. Park, J. Lee, S.-K. Kim, K. Jang, Y. Sung-Bong, A forwarding scheme based on swarm intelligence and percolation centrality in opportunistic networks, Wireless Netw. 22 (8) (2016) 2511–2521.

[33] Y. Peng, G. Wang, W. Nao, Energy-efficient transmission strategy by using optimal stopping approach for mobile networks, Mob. Inf. Syst., 2016 (2016).

[34] N.S. Albert, Optimal Stopping Rules, vol. 8, Springer Science & Business Media, 2007.

[35] A.P. Silva, S. Burleigh, C.M. Hirata, O. Katia, A survey on congestion control for delay and disruption tolerant networks, Ad Hoc Netw. 25 (2015) 480–494.

[36] C.C. Sobin, V. Raychoudhury, G. Marfia, S. Ankita, A survey of routing and data dissemination in delay tolerant networks, J. Netw.Comput. Appl. 67 (2016) 128–146.

[37] A. Socievole, E. Yoneki, F. De Rango, C. Jon, Ml-sor: message routing using multi-layer social networks in opportunistic communications, Comput. Netw. 81 (2015) 201–219.

[38] M. Tamaki., Recognizing both the maximum and the second maximum of a sequence, J. Appl. Probab. (1979) 803–812.

[39] A. Vahdat, D. Becker, et al., Epidemic routing for partially connected ad hoc networks, 2000,.

[40] E. Yoneki, P. Hui, S. Chan, C. Jon, A socio-aware overlay for publish/subscribe communication in delay tolerant networks, in: In Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems, pages 225–234., ACM, 2007.

[41] X. Zhang, X. Wang, A. Liu, Q. Zhang, T. Chaojing, Cooperation enforcement scheme based on reputation for delay tolerant networks, in: In Computer Science and Network Technology (ICCSNT), 2011 International Conference on, volume 4, pages 2372–2376., IEEE, 2011.

[42] B. Zhao, F. Vasilis, Optimal stopping for energy efficiency with delay constraints in cognitive radio networks, In Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on, pages 820–825, IEEE, 2012.

[43] D. Zheng, W. Ge, Z. Junshan, Distributed opportunistic scheduling for ad hoc networks with random access: an optimal stopping approach, IEEE Trans. Inf. Theory 55 (1) (2009) 205–222.

[44] G. Zyba, G.M. Voelker, S. Ioannidis, D. Christophe, Dissemination in opportunistic mobile ad-hoc networks: the power of the crowd, in: In INFOCOM, 2011 Proceedings IEEE, pages 1179–1187, IEEE, 2011.

Carlos Borrego Born in Madrid. He received his degree in Computer Science (6 year programme) at the Faculty of Computer Science at the Polytechnic University of Madrid. After finishing his studies he moved to work for CERN (Geneva, Switzerland). In 2001 moved to CASPUR, University La Sapienza (Rome, Italy) and stayed there for four years. In 2005 moved to the Autonomous University of Barcelona (Barcelona, Spain) where he finished his PhD and worked for Pic and Ifae research centers. He is actually researcher and adjunct professor at the Department of Information and Communications Engineering dEIC. He gives lectures on computer networks and cryptography.

Joan Borrell received his PhD in Computer Science in 1996 from Universitat Autónoma de Barcelona, where he is an associate professor and former head of the Department of Information and Communications Engineering. In 2009/10 he was a visiting researcher of the Security Group and the Mobile & Internet Systems Laboratory of University College Cork, Ireland. His current research interest is focused on opportunistic networking and its security. In the past, he had been working on cryptographic schemes for electronic voting and electronic gambling, and on mobility and security of software agents. Since 2001 he is a member of the Scientific Advisory Board of SCYTL Secure Electronic Voting.

Sergi Robles received his PhD in Computer Science from Autonomous University of Barcelona. He is an associate professor in the Department of Information and Communications Engineering at the Universitat Autónoma de Barcelona, where he leads the Security of Networks and Distributed Applications (SeNDA) research group. His latest research interests include mobile agents and security, and routing in Delay Tolerant Networks.